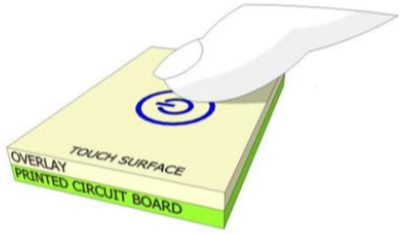
Exhibit 9

U.S. Patent No. 9,411,472 ("'472 Patent")

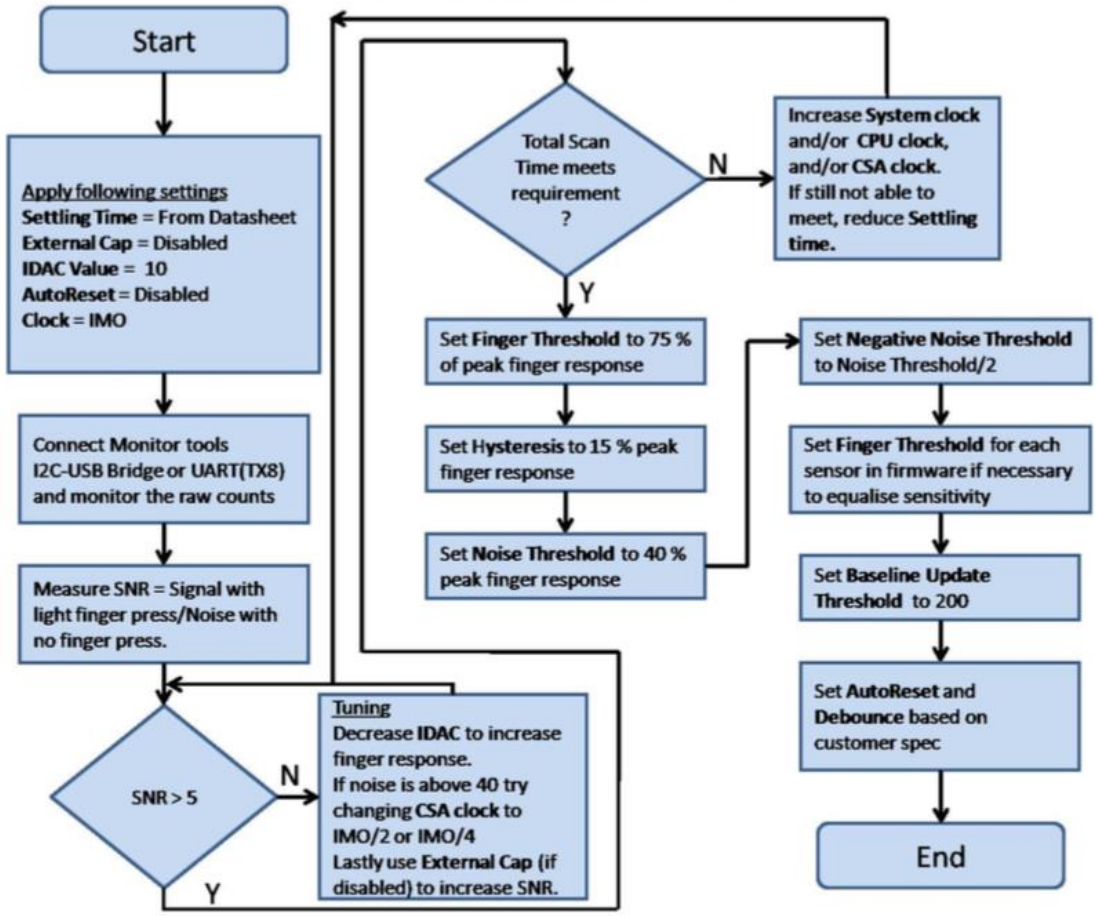
Exemplary Accused Product

Cypress products, including at least each of the following products (and their variations) infringe at least Claim 1 of the '472 Patent: Capsense enabled Cypress products, including MBR3, CY8CMBR2110, CY8CMBR2044, CY8CMBR2016, CY8CMBR2010, CY8CMBR3XXX, and Capsense-enabled PSoC. The infringement chart below is based on the Cypress PSoC with CapSense ("CapSense"), which is exemplary of the infringement of the '472 Patent.

Claim	CapSense
[1pre] A touch sensor comprising:	The CapSense touchcontroller provides capacitive touch sensing functionality, including in noisy and moist environments, and is designed to operate with a touch sensor. Cypress' CapSense controllers use changes in capacitance to detect the presence of a finger on or near a touch surface, as shown in Figure 2-1. This touch button example illustrates a capacitive sensor replacing a mechanical button. The sensing function is achieved using a combination of hardware and firmware. See the Glossary for the definitions of CapSense terms. Figure 2-1. Illustration of a Capacitive Sensor Application  *See* Getting Started with CapSense, at p. 10, https://www.cypress.com/file/41076/download Firmware is a vital component of the CapSense system. It processes the raw count data and makes logical decisions. The amount of firmware development required for your application depends on which CapSense controller family you select.

	See Getting Started with CapSense, at p. 12, https://www.cypress.com/file/41076/download Optimal CapSense system performance depends on the board layout, button dimensions, overlay material, and application requirements. In addition to these factors, switching frequency and threshold levels must be carefully selected for robust and reliable performance. Tuning is the process of determining the optimum values for these parameters. Tuning is required to maintain high sensitivity to touch and to compensate for process variations in the sensor board, overlay material, and environmental conditions. Many of the CapSense devices support SmartSense, Cypress' Auto-tuning algorithm, which automatically sets parameters for optimal performance and continuously compensates for system, manufacturing and environmental changes. See SmartSense Auto-Tuning for more information. *See* Getting Started with CapSense, at p. 18, https://www.cypress.com/file/41076/download PSoC uses Cypress patented capacitive touch sensing methods known as CapSense Sigma Delta (CSD) for self-capacitance sensing and CapSense Crosspoint (CSX) for mutual-capacitance scanning. The CSD and CSX touch sensing methods provide the industry's best-in-class Signal-to-Noise Ratio. These sensing methods are a combination of hardware and firmware techniques. *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 15, https://www.cypress.com/file/46081/download Tuning the touch sensing user interface is a critical step in ensuring proper system operation and a pleasant user experience. The typical design flow involves tuning the sensor interface in the initial design phase, during system integration, and finally production fine-tuning before the production ramp. Because tuning is an iterative process, it can be time-consuming. SmartSense Auto-Tuning helps to simplify the user interface development cycle. In addition, the method is easy to use and reduces the design cycle time by eliminating the tuning process throughout the product development cycle, from prototype to mass production. *See* Getting Started with CapSense, at p. 20, https://www.cypress.com/file/41076/download

Figure 8. CSA Calibration Flowchart



See CSA Software Filters with Ezl2Cs Slave on CY8C20xx6, at p. 10, https://www.cypress.com/file/137541/download

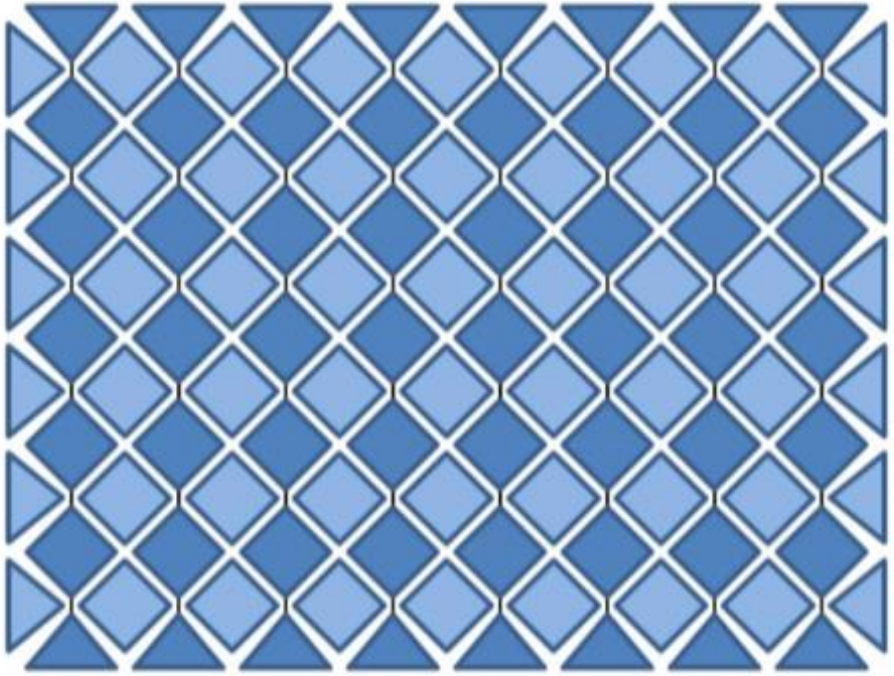
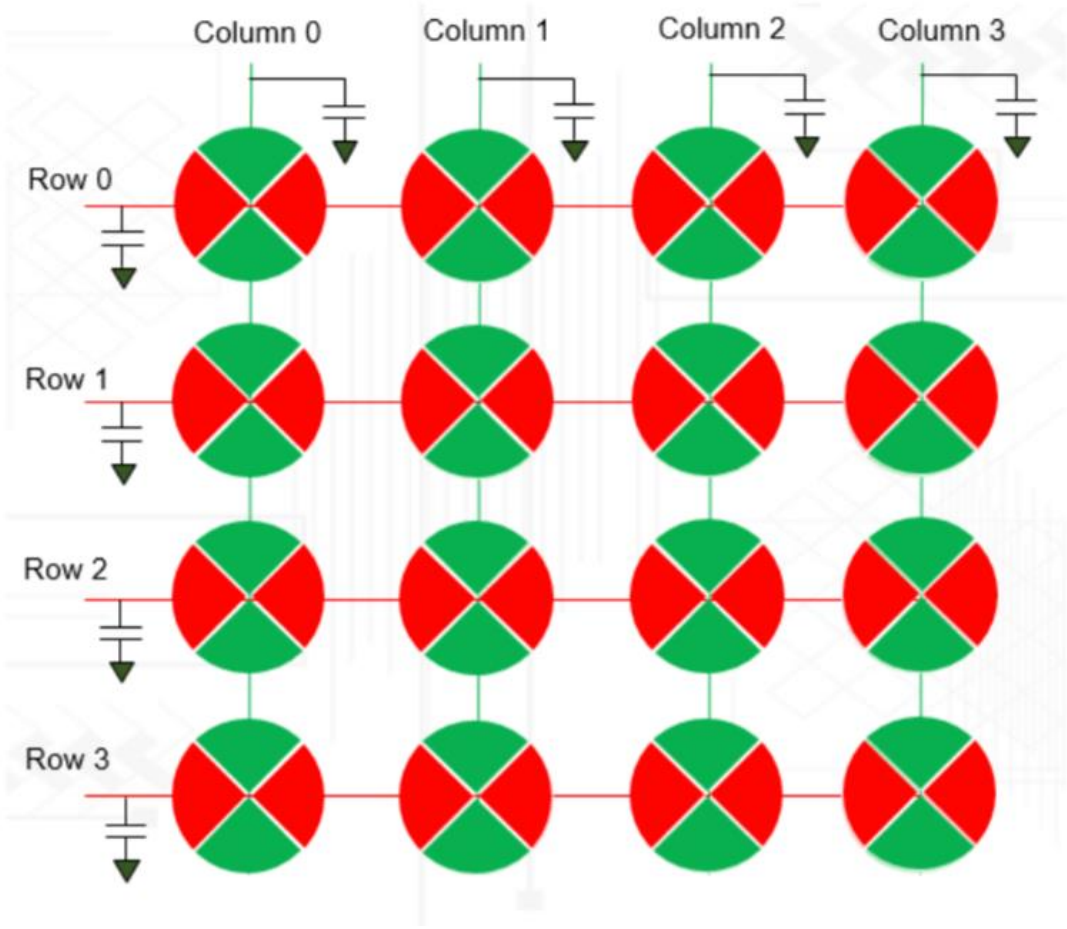
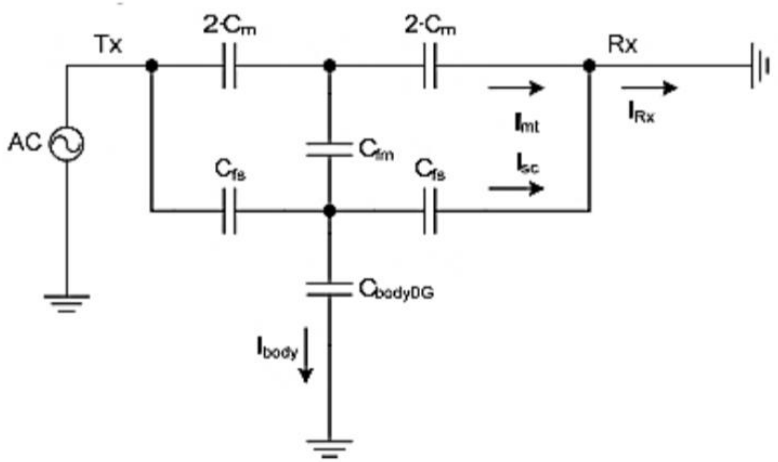
[1a] a plurality of sense electrodes; and	Touch sensors for CapSense have a plurality of sense electrodes, which can be in different configurations. For example, a CapSense-enabled touch sensor is a matrix touch sensor that has a plurality of sense electrodes. Alternatively, the touch sensor may be an array of capacitive-type buttons having a plurality of sense electrodes. Figure 2-27. Trackpad Sensor Arrangement  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 27, https://www.cypress.com/file/41076/download

Figure 2-14. Matrix Buttons Based on CSD



See PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 25, https://www.cypress.com/file/41076/download

	In a mutual-capacitance measurement system, a digital voltage (signal switching between V_{DDD} and GND) is applied to the TX pin and the amount of charge received on the RX pin is measured. The amount of charge received on the RX electrode is directly proportional to the mutual capacitance (C_M) between the two electrodes. *See* Getting Started with CapSense, at p. 14, https://www.cypress.com/file/41076/download Figure 7-34. Equivalent Circuit of the CSX Sensor when Finger Is Placed on the Button  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 135, https://www.cypress.com/file/46081/download
[1b] a controller communicatively coupled to the plurality of sense electrodes, the controller configured to:	The CapSense touchcontroller is communicatively coupled to the plurality of sense electrodes. *See* limitation [1a].

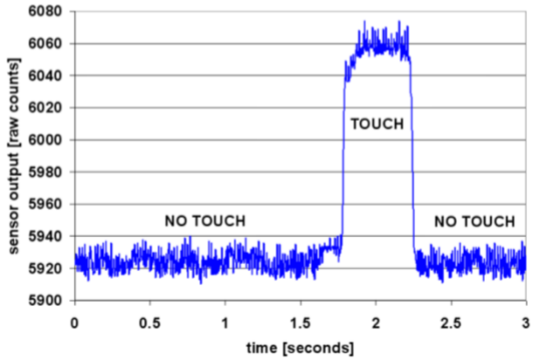
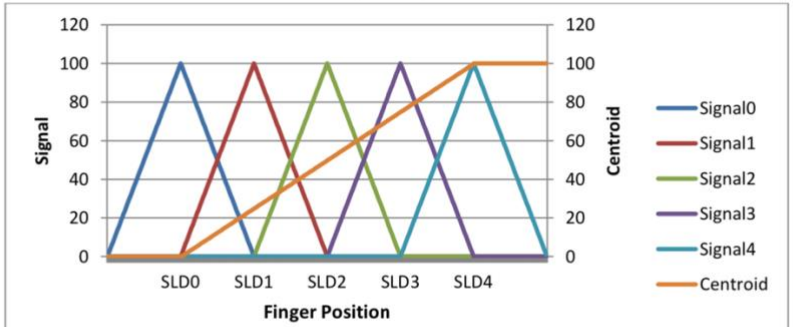
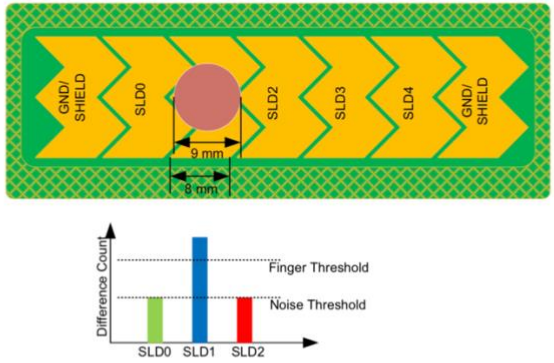
[1c] receive a plurality of signals from the plurality of sense electrodes associated with an interaction with the touch sensor by an external object, the plurality of signals indicative of an amount of capacitance between the touch sensor and the external object;	The CapSense touchcontroller receives a plurality of signals from the plurality of sense electrodes associated with an interaction with the touch sensor by an external object, the plurality of signals indicative of an amount of capacitance between the touch sensor and the external object. For example, a plurality of signals indicative of capacitance is received by the CapSense touchcontroller when a finger touches the touch sensor. Figure 2-13 shows a plot of raw count over time. When a finger touches the sensor, C_M decreases from C_M to C^1_M (see Figure 2-10) hence the counter output decreases. The firmware normalizes the raw count such that the raw counts go high when C_M decreases. This is to maintain the same visual representation of raw count between CSD and CSX methods. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present). Figure 2-13. Raw Count versus Time  https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 16.

Figure 3-52. Ideal Slider Segment Signals and Centroid Response



Here, Signalx = Difference Count - Noise Threshold value for segment x.

Figure 3-53. Ideal Slider Signals



Equation 3-21. Segment width and air-gap relation with finger diameter

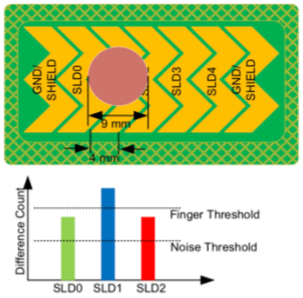
$$W + 2A = finger\ diameter$$

Typically, an average human finger diameter is approximately 9 mm. Based on this average finger diameter and Equation 3-21, the recommended slider-segment-width and air-gap is 8 mm and 0.5 mm respectively.

If the *slider-segment-width + 2 * air-gap* is lesser than *finger diameter*, as required per Equation 3-21, the centroid response will be non-linear. This is because, in this case, a finger placed on the slider will add capacitance, and hence valid signal to more than two slider-segments at some given position, as Figure 3-54 shows. Thus, calculated centroid position per Equation 3-21 will be non-linear, as Figure 3-54 shows.

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 76.

Figure 3-54. Finger Causes Valid Signal on More Than Two Segments when Slider Segment Width Is Lower Than Recommended



Equation 3-22. Centroid algorithm used by CapSense

$$Centroid\ position = \left(\frac{S_{x+1} - S_{x-1}}{S_{x+1} + S_{x0} + S_{x-1}} + maximum\right) * \frac{Resolution}{(n-1)}$$

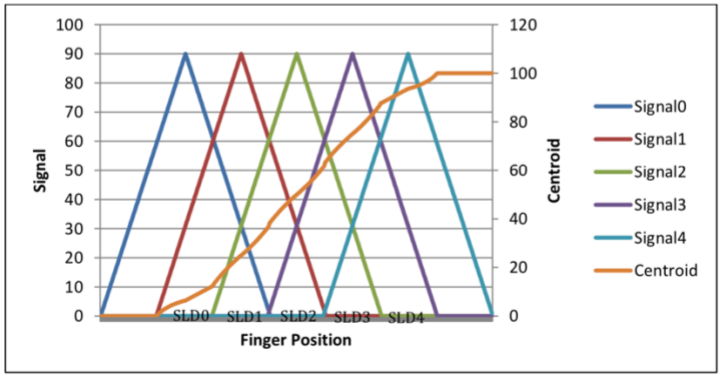
Resolution – API Resolution set in the Customizer,

n – Number of sensor elements in the Customizer.

maximum: Index of element which gives maximum signal.

Si – different counts (with subtracted Noise Threshold value) near by the maximum position

Figure 3-55. Nonlinear Centroid Response when Slider Segment Width Is Lower Than Recommended

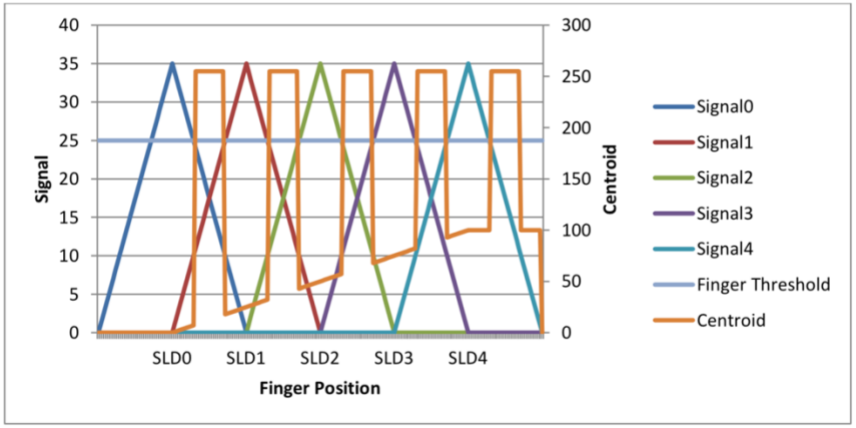


Note that even though a *slider-segment-width* value of less than *finger diameter – 2 * air-gap* provides a non-linear centroid response, as Figure 3-54 shows; it may still be used in an end application where the linearity of reported centroid versus actual finger position does not play a significant role. However, a minimum value of slider-segment-width must be maintained, based on overlay thickness, such that, at any position on the effective slider length, at-least one slider-segment provides an SNR of >=5:1 (i.e. signal >= Finger Threshold parameter) at that position. If the slider-

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 77.

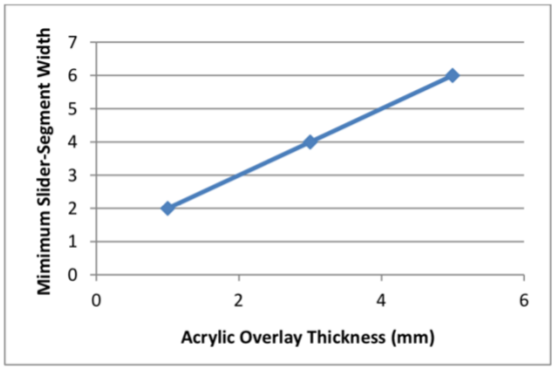
segment-width is too low, a finger may not be able to couple enough capacitance, and hence, none of the slider-segments will have a 5:1 SNR, resulting in a reported centroid value of 0xFF[8], as Figure 3-56 shows.

Figure 3-56. Incorrect Centroid Reported when Slider Segment Width Is Too Low



The minimum value of slider-segment-width for certain specific overlay thickness values, for an acrylic overlay, are provided in Table 3-10. For acrylic overlays of thickness values, which are not specified in Table 3-10 and Figure 3-57 may be used to estimate the minimum slider-segment-width.

Figure 3-57. Minimum Slider-Segment-Width w.r.t. Overlay Thickness for an Acrylic Overlay



If the *slider-segment-width + 2 * air-gap* is higher than *finger diameter*, as required per Equation 3-21, the centroid response will have flat spots i.e., if the finger is moved a little near the middle of any segment, the reported centroid position will remain constant as Figure 3-58 shows. This is because, as Figure 3-59 shows, when the finger is placed in the middle of a slider-segment, it will add valid signal only to that segment even if the finger is moved a little towards the adjacent segments.

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 78.

Figure 3-58. Flat Spots (Nonresponsive Centroid) when Slider Segment Width Is Higher Than Recommended

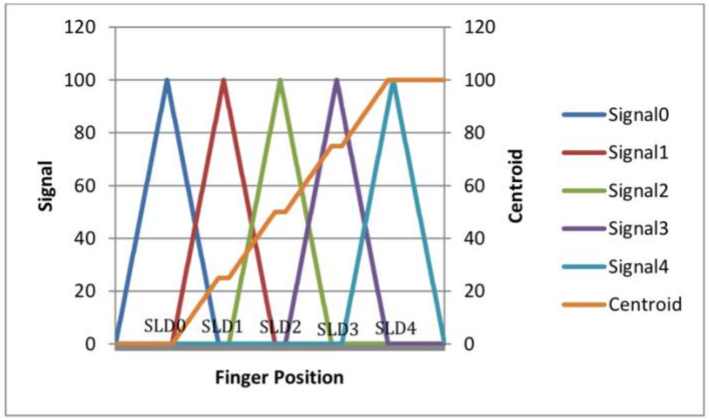
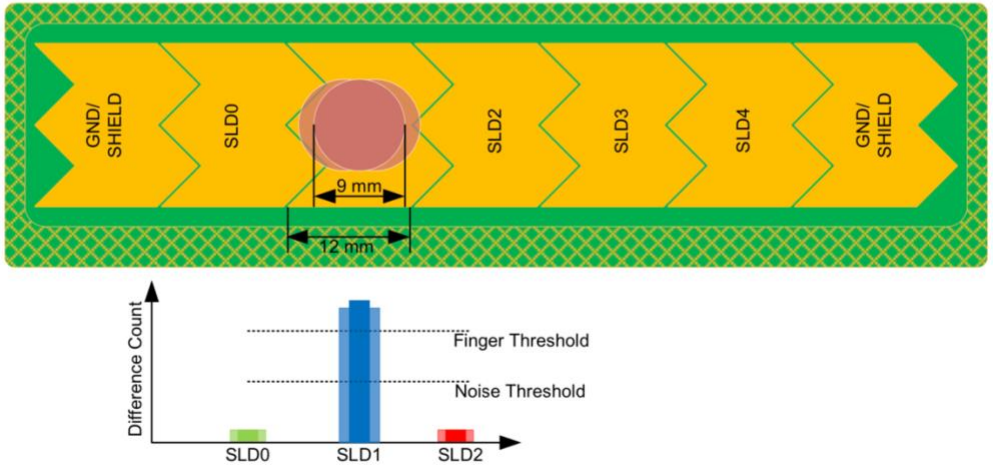


Figure 3-59. Signal on Slider Segments when Slider Segment Width Is Higher Than Recommended



Note that if the *slider-segment-width + 2 * air-gap* is higher than *finger diameter,* it may be possible to increase and adjust the sensitivity of all the slider segments such that even if the finger is placed in middle of a slider-segment, the adjacent sensors report a difference count value equal to noise threshold value (as required per

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 79.

2.3.1 CapSense Sigma Delta Modulator (CSD) Sensing Method

Figure 2-11 shows a simplified block diagram of the CSD method.

In CSD, each GPIO has a switched-capacitance circuit that converts the sensor capacitance into an equivalent current. An analog multiplexer then selects one of the currents and feeds it into the current to digital converter. The current to digital converter is similar to a sigma delta ADC. The output count of the current to digital converter, known as **raw count**, is a digital value that is proportional to the self-capacitance between the electrodes.

Equation 2-3. Raw Count and Sensor Capacitance Relationship in CSD

$$\text{raw count} = G_C C_S$$

Where G_C is the capacitance to digital conversion gain of CSD, and

C_S is the self-capacitance of the electrode.

Figure 2-11. Simplified Diagram of CapSense Sigma Delta Method

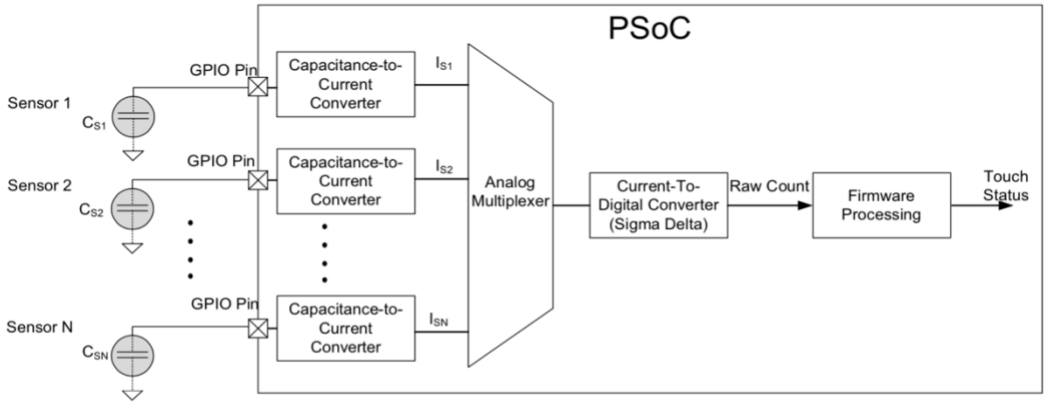


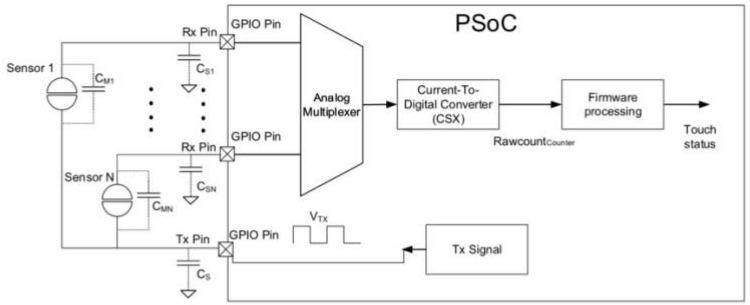
Figure 2-13 shows a plot of raw count over time. When a finger touches the sensor, the C_S increases from C_P to C_P + C_F, and the raw count increases. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present).

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 15.

2.3.2 CapSense Crosspoint (CSX) Sensing Method

Figure 2-12 shows the simplified block diagram of the CSX method.

Figure 2-12. Simplified Diagram of CSX Method



With CSX, a voltage on the Tx pin (or Tx electrode) couples charge on to the RX pin. This charge is proportional to the mutual capacitance between the Tx and Rx electrodes. An analog multiplexer then selects one of the Rx channel and feeds it into the current to digital converter.

The output count of the current to digital converter, known as $\mathbf{Rawcount_{Counter}}$, is a digital value that is proportional to the mutual-capacitance between the Rx and Tx electrodes as shown by Equation 2-4.

Equation 2-4. Raw Count and Sensor Capacitance Relationship in CSX

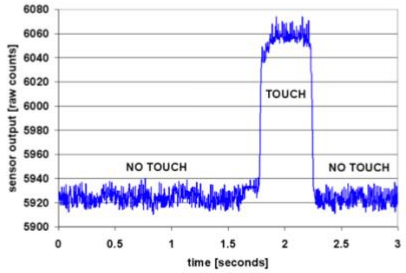
$$\mathrm{Rawcount_{Counter}} = G_{CM}\, C_M$$

Where G_{CM} is the capacitance to digital conversion gain of Mutual Capacitance method, and

C_M is the mutual-capacitance between two electrodes.

Figure 2-13 shows a plot of raw count over time. When a finger touches the sensor, C_M decreases from C_M to C^1_M (see Figure 2-10) hence the counter output decreases. The firmware normalizes the raw count such that the raw counts go high when C_M decreases. This is to maintain the same visual representation of raw count between CSD and CSX methods. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present).

Figure 2-13. Raw Count versus Time



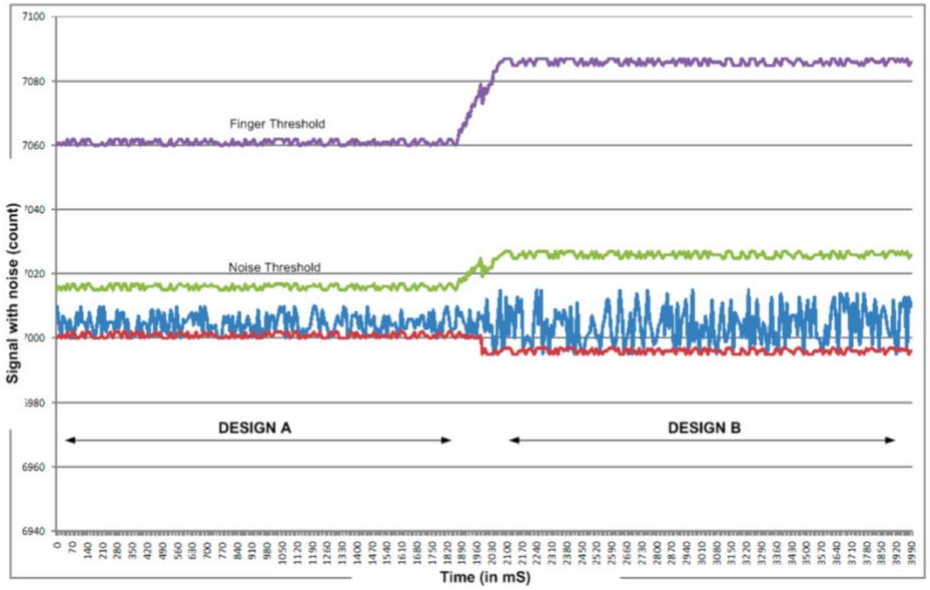
https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 16.

[1d] access a stored threshold value, the threshold value indicating a threshold magnitude of capacitance; determine a grounding status of the touch sensor based on a strength of a charge return path between the touch sensor and a ground; adjust the stored threshold value based on the determined grounding status of the touch sensor;	The CapSense touchcontroller accesses a stored threshold value, the threshold value indicating a threshold magnitude of capacitance, determines a grounding status of the touch sensor based on a strength of a charge return path between the touch sensor and a ground, and adjusts the stored threshold value based on the determined grounding status of the touch sensor. For example, the SmartSense Auto-Tuning dynamically tunes the noise threshold and finger touch thresholds based on different parameters, including the grounding status of the touch sensor. 2.5.2 SmartSense Auto-Tuning *2.5.2.1 What Is SmartSense?* Tuning the touch sensing user interface is a critical step in ensuring proper system operation and a pleasant user experience. The typical design flow involves tuning the sensor interface in the initial design phase, during system integration, and finally production fine-tuning before the production ramp. Because tuning is an iterative process, it can be time-consuming. SmartSense Auto-Tuning helps to simplify the user interface development cycle. In addition, the method is easy to use and reduces the design cycle time by eliminating the tuning process throughout the product development cycle, from prototype to mass production. *2.5.2.2 What Does SmartSense Do?* SmartSense tunes each CapSense sensor automatically at power-up and then monitors and maintains optimum sensor performance during runtime. The number of parameters to be tuned is reduced from 17 in CSD to 4 with SmartSense. ■ **Power-up tuning:** SmartSense tunes the parameters of each sensor based on the individual sensor parasitic capacitance to get the desired sensitivity for the sensor. ■ **Runtime tuning:** Noise in the system is measured dynamically. The thresholds are adjusted accordingly for each sensor to overcome false triggering due to dynamic variations in noise in the CapSense system. *2.5.2.3 How and Where is SmartSense Helpful?* SmartSense technology adapts for manufacturing variations in PCBs, overlays, and noise generators, such as LCD inverters, AC line noise, and switch-mode power supplies and automatically tunes them out. SmartSense handles changes in system environment, such as temperature, humidity, and noise sources such as RF, SMPS, LCD Inverter, and AC line noise. The following sections describe scenarios in which SmartSense is instrumental in adapting to the external noise. By maintaining a robust signal-to-noise ratio, the false triggering of buttons is prevented. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 20.

2.5.2.3.1 Different Noise Levels in Different Designs

SmartSense technology dynamically tunes itself (adjusts noise and finger thresholds) for different noise environments. In Figure 2-16, Design A and Design B have different noise levels. To maintain a minimum SNR of 5:1, you must adjust the dynamic threshold. SmartSense does this automatically, allowing seamless transition from one model to another with minimal or no tuning required.

Figure 2-16. Different Noise Levels in Design A and B Being Compensated Automatically

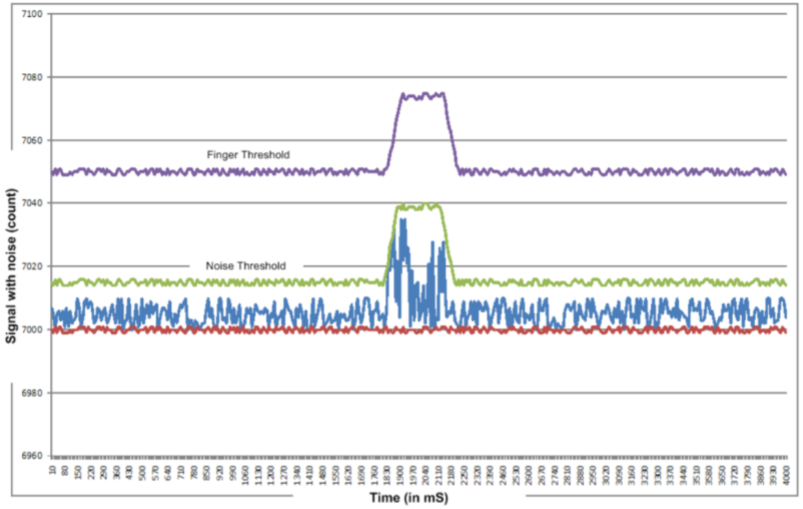


https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 20.

2.5.2.3.2 Noise Spikes During Production

SmartSense technology also automatically tunes out the noise spikes (in production) that may not be seen during the design stage, as indicated in Figure 2-17. This is a powerful SmartSense feature that prevents false button presses in the end system, which prevents a failure analysis for a mass production design.

Figure 2-17. Finger Threshold Dynamically Adjusted to Prevent False Button Touches



2.5.2.4 When Is Manual-Tuning Advantageous?

SmartSense allows a device to calibrate itself for optimal performance and complete the entire tuning process automatically. This technology will meet the needs of most designs, but, in the case where SmartSense will not work or there are specific SNR or power requirements, the CapSense CSD parameters can be manually adjusted to meet system requirements. This is called manual tuning. Some advantages of manual tuning, as opposed to SmartSense Auto-tuning are:

- Strict control over parameter settings: SmartSense sets all of the parameters automatically. However, there may be situations where you need strict control over the parameters. For example, use manual tuning if you need to strictly control the time CSD takes to scan a group of sensors. This can be done to reduce EMI in systems.
- Supports higher parasitic capacitances: SmartSense supports parasitic capacitances as high as 45 pF for a 0.2-pF finger capacitance, and as high as 35 pF for a 0.1-pF finger capacitance. If the parasitic capacitance is higher than the value supported by SmartSense, use manual tuning.

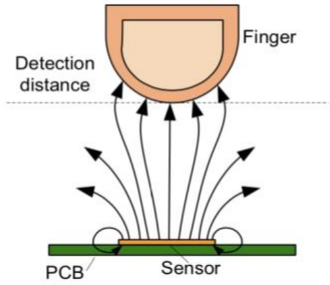
See the device-specific Design Guide for the step-by-step procedure on manual tuning.

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 21.

- **Nearby floating or grounded conductive objects:** The proximity-sensing distance reduces drastically if there is any floating or grounded conductive object nearby. The following factors cause the proximity-sensing distance to reduce drastically when conductive objects are placed close to the proximity sensor:
  - ☐ The C_P of the sensor increases. Larger sensor C_P often requires reducing the sensor switching frequency, causing the proximity-sensing distance to decrease.
  - ☐ A grounded conductive object catches a part of the sensor electric field and reduces the capacitance added by the target as shown in Figure 3-41.

  You should either remove the nearby conductive object or use a shield electrode to isolate the proximity sensor from the conductive object. The influence of a nearby metal surface on the proximity sensor is reduced by placing a shield electrode between the proximity sensor and the metal object, as shown in Figure 3-42. For layout recommendations on shield electrode, see the Shield Electrode and Guard Sensor section.

Figure 3-40. Electrical Field Propagation for a Single Sensor Configuration without a Metal Object



https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 68.

Figure 3-41. Electrical Field Propagation for a Single Sensor Configuration with a Solid Metal Object

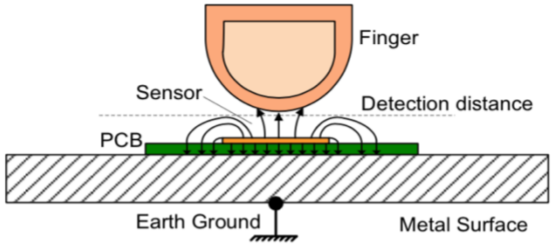
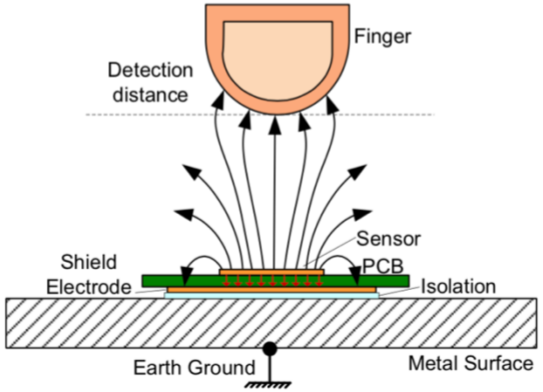
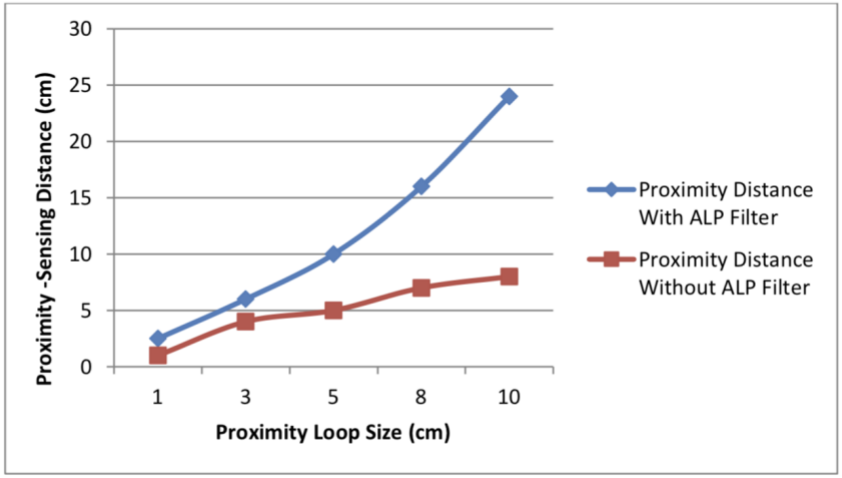


Figure 3-42. Using a Shield Electrode to Decrease the Metal Object's Influence



https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 69.

	3.8.14.1 Shield Electrode for Proximity Sensing If you want to use shield electrode for reducing sensor C_P or reduce the effect of nearby floating/grounded conductive objects or provide directionality to proximity sensing, follow the below guidelines: ■ To reduce the sensor C_P, use a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in the top layer and a hatch fill of 0.17 mm (7 mil) trace and 1.778 mm (70 mil grid) in the bottom layer and drive it with driven shield signal. ■ To reduce the effect of floating/grounded conductive object on the proximity-sensing distance, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the conductive object and drive the hatch fill with the driven shield signal. ■ To make the proximity sensing unidirectional, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the direction in which proximity detection should be avoided and drive the hatch fill with the driven shield signal. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 88. 3.3.1.1.1 Ground Plane In general, a proper ground plane on the PCB reduces both RF emissions and interference. However, solid grounds near CapSense sensors or traces connecting these sensors to the PSoC pins increase the parasitic capacitance of the sensors. It is thus recommended that you use hatched ground planes surrounding the sensor and on the bottom layer of the PCB, below the sensors, as explained in the Ground Plane section. A solid ground may be used below the device and other circuitry on the PCB, which is far from CapSense sensors and traces. A solid ground flood is not recommended within 10 mm of CapSense sensors or traces. Multiple-layer boards should be the preferred choice. If you are using a board with four layers or more, you can provide a complete layer for ground that will further help to reduce emissions as it reduces the ground bounce significantly. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 44.

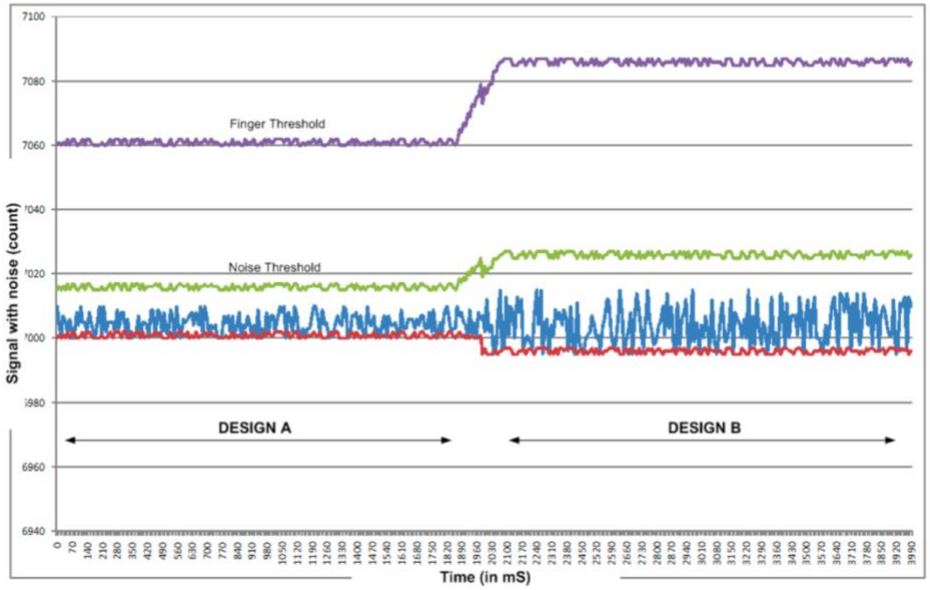
	■ **Size of the sensor:** The proximity sensor size depends on various factors, such as the required proximity-sensing distance, presence of noise sources, and floating or grounded conductive objects. Noise sources and floating or grounded conductive objects reduce the SNR and the proximity-sensing distance. Therefore, large proximity sensors are needed to achieve the proximity-sensing distance required in your design. Figure 3-38 shows the relationship between the proximity-sensor loop size and the proximity-sensing distance for a given system. A larger sensor area results in more electric field lines coupling with the target object, resulting in increase in the sensor signal. However, a large sensor area results in a high sensor C_P and high noise, and thus reduces the proximity-sensing distance. Using a loop sensor (Figure 3-37 (b)) instead of a solid-fill sensor results in a low sensor C_P, low noise, and thus a large proximity-sensing distance. Also, loop sensors require less sensor area, leaving more space to place components on the PCB. Figure 3-38. Proximity Loop Size versus Proximity Distance  https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 67.
[1e] after adjusting the stored threshold value, determine whether to process the interaction as a touch by the external object based on a comparison of the amount of capacitance with the adjusted threshold value; and after adjusting the stored threshold value based on the determined grounding status of the touch	The CapSense touchcontroller, after adjusting the stored threshold value, determines whether to process the interaction as a touch by the external object based on a comparison of the amount of capacitance with the adjusted threshold value, and after adjusting the stored threshold value based on the determined grounding status of the touch sensor, determines that the external object has not touched the touch sensor within a predetermined amount of time and changes the stored threshold value back to an original value, the original value comprising a value of the stored threshold value before it was adjusted based on the determined grounding status of the touch sensor.

sensor: determine that the external object has not touched the touch sensor within a predetermined amount of time; and change the stored threshold value back to an original value, the original value comprising a value of the stored threshold value before it was adjusted based on the determined grounding status of the touch sensor.	For example, after the SmartSense Auto-Tuning dynamically tunes the noise threshold and finger touch thresholds based on different parameters, including the grounding status of the touch sensor, the SmartSense Auto-Tuning functionality reverts back to a default or previous state having the default or previous noise and touch thresholds. 2.5.2 SmartSense Auto-Tuning *2.5.2.1 What Is SmartSense?* Tuning the touch sensing user interface is a critical step in ensuring proper system operation and a pleasant user experience. The typical design flow involves tuning the sensor interface in the initial design phase, during system integration, and finally production fine-tuning before the production ramp. Because tuning is an iterative process, it can be time-consuming. SmartSense Auto-Tuning helps to simplify the user interface development cycle. In addition, the method is easy to use and reduces the design cycle time by eliminating the tuning process throughout the product development cycle, from prototype to mass production. *2.5.2.2 What Does SmartSense Do?* SmartSense tunes each CapSense sensor automatically at power-up and then monitors and maintains optimum sensor performance during runtime. The number of parameters to be tuned is reduced from 17 in CSD to 4 with SmartSense. ■ **Power-up tuning:** SmartSense tunes the parameters of each sensor based on the individual sensor parasitic capacitance to get the desired sensitivity for the sensor. ■ **Runtime tuning:** Noise in the system is measured dynamically. The thresholds are adjusted accordingly for each sensor to overcome false triggering due to dynamic variations in noise in the CapSense system. *2.5.2.3 How and Where is SmartSense Helpful?* SmartSense technology adapts for manufacturing variations in PCBs, overlays, and noise generators, such as LCD inverters, AC line noise, and switch-mode power supplies and automatically tunes them out. SmartSense handles changes in system environment, such as temperature, humidity, and noise sources such as RF, SMPS, LCD Inverter, and AC line noise. The following sections describe scenarios in which SmartSense is instrumental in adapting to the external noise. By maintaining a robust signal-to-noise ratio, the false triggering of buttons is prevented. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 20.

2.5.2.3.1 Different Noise Levels in Different Designs

SmartSense technology dynamically tunes itself (adjusts noise and finger thresholds) for different noise environments. In Figure 2-16, Design A and Design B have different noise levels. To maintain a minimum SNR of 5:1, you must adjust the dynamic threshold. SmartSense does this automatically, allowing seamless transition from one model to another with minimal or no tuning required.

Figure 2-16. Different Noise Levels in Design A and B Being Compensated Automatically

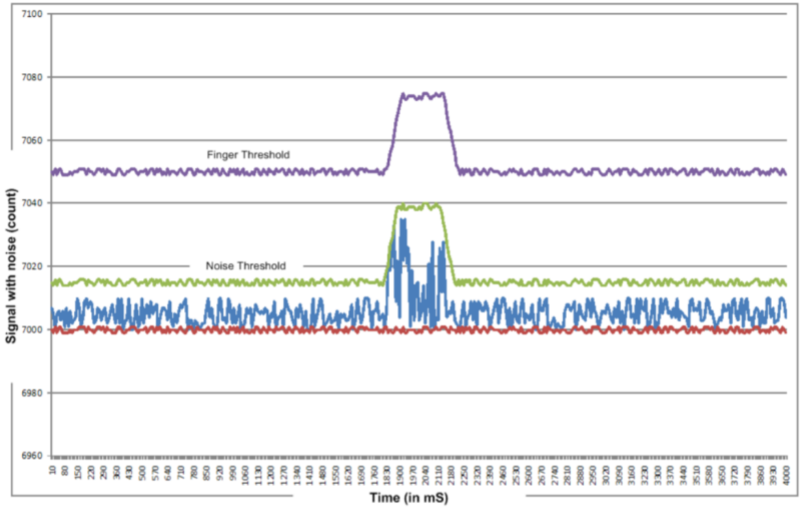


https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 20.

2.5.2.3.2 Noise Spikes During Production

SmartSense technology also automatically tunes out the noise spikes (in production) that may not be seen during the design stage, as indicated in Figure 2-17. This is a powerful SmartSense feature that prevents false button presses in the end system, which prevents a failure analysis for a mass production design.

Figure 2-17. Finger Threshold Dynamically Adjusted to Prevent False Button Touches



2.5.2.4 When Is Manual-Tuning Advantageous?

SmartSense allows a device to calibrate itself for optimal performance and complete the entire tuning process automatically. This technology will meet the needs of most designs, but, in the case where SmartSense will not work or there are specific SNR or power requirements, the CapSense CSD parameters can be manually adjusted to meet system requirements. This is called manual tuning. Some advantages of manual tuning, as opposed to SmartSense Auto-tuning are:

- Strict control over parameter settings: SmartSense sets all of the parameters automatically. However, there may be situations where you need strict control over the parameters. For example, use manual tuning if you need to strictly control the time CSD takes to scan a group of sensors. This can be done to reduce EMI in systems.
- Supports higher parasitic capacitances: SmartSense supports parasitic capacitances as high as 45 pF for a 0.2-pF finger capacitance, and as high as 35 pF for a 0.1-pF finger capacitance. If the parasitic capacitance is higher than the value supported by SmartSense, use manual tuning.

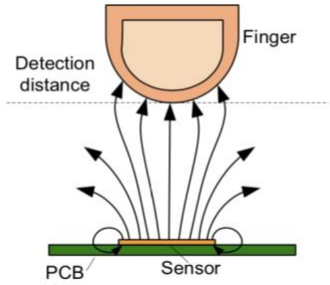
See the device-specific Design Guide for the step-by-step procedure on manual tuning.

https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 21.

- **Nearby floating or grounded conductive objects:** The proximity-sensing distance reduces drastically if there is any floating or grounded conductive object nearby. The following factors cause the proximity-sensing distance to reduce drastically when conductive objects are placed close to the proximity sensor:
  - ☐ The C_P of the sensor increases. Larger sensor C_P often requires reducing the sensor switching frequency, causing the proximity-sensing distance to decrease.
  - ☐ A grounded conductive object catches a part of the sensor electric field and reduces the capacitance added by the target as shown in Figure 3-41.

  You should either remove the nearby conductive object or use a shield electrode to isolate the proximity sensor from the conductive object. The influence of a nearby metal surface on the proximity sensor is reduced by placing a shield electrode between the proximity sensor and the metal object, as shown in Figure 3-42. For layout recommendations on shield electrode, see the Shield Electrode and Guard Sensor section.

Figure 3-40. Electrical Field Propagation for a Single Sensor Configuration without a Metal Object



https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 68.

Figure 3-41. Electrical Field Propagation for a Single Sensor Configuration with a Solid Metal Object

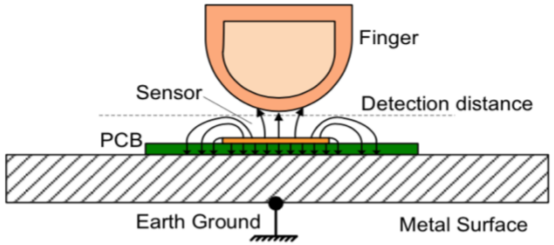
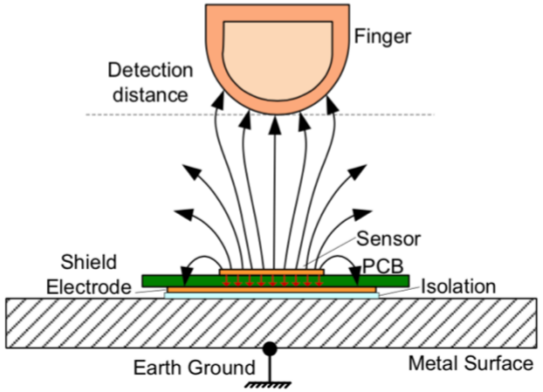


Figure 3-42. Using a Shield Electrode to Decrease the Metal Object's Influence



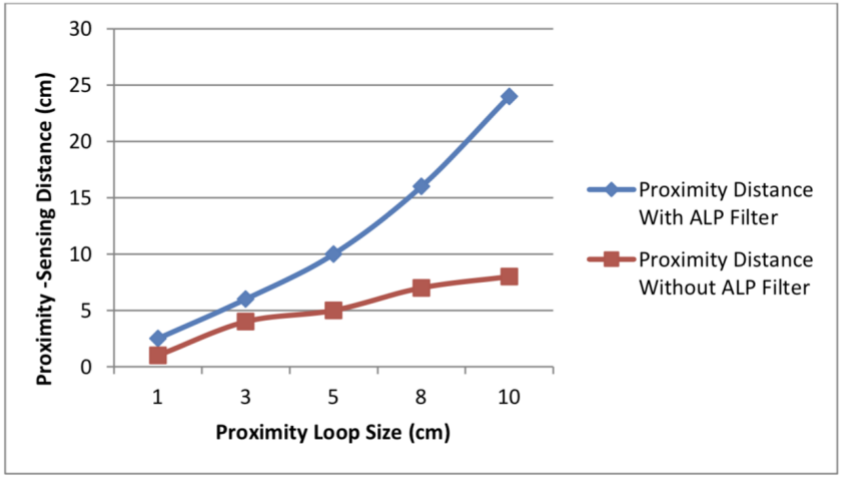
https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 69.

	3.8.14.1 Shield Electrode for Proximity Sensing If you want to use shield electrode for reducing sensor C_P or reduce the effect of nearby floating/grounded conductive objects or provide directionality to proximity sensing, follow the below guidelines: ■ To reduce the sensor C_P, use a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in the top layer and a hatch fill of 0.17 mm (7 mil) trace and 1.778 mm (70 mil grid) in the bottom layer and drive it with driven shield signal. ■ To reduce the effect of floating/grounded conductive object on the proximity-sensing distance, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the conductive object and drive the hatch fill with the driven shield signal. ■ To make the proximity sensing unidirectional, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the direction in which proximity detection should be avoided and drive the hatch fill with the driven shield signal. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 88. 3.3.1.1.1 Ground Plane In general, a proper ground plane on the PCB reduces both RF emissions and interference. However, solid grounds near CapSense sensors or traces connecting these sensors to the PSoC pins increase the parasitic capacitance of the sensors. It is thus recommended that you use hatched ground planes surrounding the sensor and on the bottom layer of the PCB, below the sensors, as explained in the Ground Plane section. A solid ground may be used below the device and other circuitry on the PCB, which is far from CapSense sensors and traces. A solid ground flood is not recommended within 10 mm of CapSense sensors or traces. Multiple-layer boards should be the preferred choice. If you are using a board with four layers or more, you can provide a complete layer for ground that will further help to reduce emissions as it reduces the ground bounce significantly. https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 44.

- **Size of the sensor:** The proximity sensor size depends on various factors, such as the required proximity-sensing distance, presence of noise sources, and floating or grounded conductive objects. Noise sources and floating or grounded conductive objects reduce the SNR and the proximity-sensing distance. Therefore, large proximity sensors are needed to achieve the proximity-sensing distance required in your design.

  Figure 3-38 shows the relationship between the proximity-sensor loop size and the proximity-sensing distance for a given system. A larger sensor area results in more electric field lines coupling with the target object, resulting in increase in the sensor signal. However, a large sensor area results in a high sensor C_P and high noise, and thus reduces the proximity-sensing distance. Using a loop sensor (Figure 3-37 (b)) instead of a solid-fill sensor results in a low sensor C_P, low noise, and thus a large proximity-sensing distance. Also, loop sensors require less sensor area, leaving more space to place components on the PCB.

Figure 3-38. Proximity Loop Size versus Proximity Distance



https://www.cypress.com/documentation/application-notes/an64846-getting-started-capsense at 67.